93-01771/

National Defence / Défense nationale

DESCRIPTION OF A DATA ACQUISITION SYSTEM FOR THE MEASUREMENTS OF EMP TRANSIENT FIELDS

by

Marc Dion
Nuclear Effects Section
Electronics Division

DEFENCE RESEARCH ESTABLISHMENT OTTAWA
TECHNICAL NOTE 92-7

PCN
041LT

March 1992
Ottawa

ABSTRACT

This technical note describes a data acquisition system developed at DREO for the measurements of very fast transient electromagnetic fields generated during EMP testing. The system consists of several high speed transient digitizers, all connected to a personal computer. It has the capability to simultaneously record on several channels the response of single-shot events. The overall system is controlled by a master program called DAQ. DAQ can control a number of digitizers simultaneously, allowing automation of part of the measurement process. The data transferred to the PC is automatically bound with related information such as: experiment description, setup information (sensor, cable, attenuator, and instrument used), digitizer parameters, and data processing. DAQ can archive and retrieve the results of a large number of experiments. DAQ can produce plots of the results on the screen or on hardcopy, and also has the capability to import the plots into popular word processors for the preparation of reports. DAQ incorporates several digital signal processing routines, which may be used to process the raw data. Various implementations of Finite Impulse Response (FIR) and Infinite Impulse Response (IIR) filters are supported, as well as some of the classical approaches such as Butterworth low-pass and hi-pass filters.

RÉSUMÉ

Cette note technique décrit un système d'acquisition de données développé au CRDO pour permettre les mesures de champ électromagnétique extrêmement rapide qui sont générés lors des test d'IEM. Le système comprends plusieurs numériseurs très rapides reliés à un ordinateur personnel, permettant de contrôler plusieurs canaux simultanément et de permettre une certaine automatisation des opérations. Toutes les données obtenues des instruments sont combinées avec d'autres informations, telles que: une description de l'expérience, une description de l'arrangement des capteurs, câbles, atténuateurs et instruments utilisés, les paramètres des instruments, ainsi qu'une description des traitements numériques à effectués. Le système permet de gérer sur disque une large collection d'expériences. Il peut aussi afficher les résultats sous forme graphique à l'écran ou sur papier, ou encore sous forme compatible avec plusieurs logiciels de traitement de texte. Le système permet de définir des filtres numériques FIR ou IIR, de même que les approches classiques des filtres passe-bas ou passe-haut, telles que les filtres Butterworth.

EXECUTIVE SUMMARY

DREO has built a large experimental facility for the measurement of the effects of nuclear electromagnetic pulse (EMP) on systems. The transient field generated is extremely short and difficult to measure. This technical note describes the data acquisition system designed to collect and archive these measurements.

The system consists of several high speed transient digitizers, all connected to a personal computer through an IEEE-488 instrumentation bus. It has the capability to simultaneously record on several channels the response of single-shot events. The overall system is controlled by a master program developed at DREO. This program, called DAQ, runs in an interactive environment, allowing one to easily manipulate the data and create new routines.

DAQ has the capability of controlling a number of digitizers simultaneously, allowing one to automate part of the measurement process. As the data is transferred to the PC, it is automatically bound with related information such as: experiment description, setup information (sensor, cable, attenuator, and instrument used), digitizer parameters, and data processing. The system is designed to easily archive and retrieve the results of a large number of experiments. DAQ can produce plots of the results on the screen or on hardcopy, and also has the capability to import them into popular word processors for the preparation of reports.

DAQ has an extensive set of built-in digital signal processing routines, which are used to enhance signals, compensate for errors, recover the transfer functions of sensors and cables, or simply for data analysis. Various implementations of FIR and IIR filters are supported, as well as some of the classical approaches such as Butterworth low-pass and hi-pass filters.

TABLE OF CONTENTS

PAGE

ABSTRACT iii

EXECUTIVE SUMMARY v

TABLE OF CONTENTS vii

1.0 INTRODUCTION 1
- 1.1 INSTRUMENTATION 1
- 1.2 SYSTEM DESCRIPTION 1
- 1.3 SYSTEM CONFIGURATION 2
  - 1.3.1 CONFIGURING DOS 2
  - 1.3.2 CONFIGURING ASYST 3
- 1.4 LOADING THE DAQ SOFTWARE 4
- 1.5 MULTI-TASKING WITH DESQVIEW 4

2.0 DATA ACQUISITION PACKAGE 6
- 2.1 INSTRUMENTATION CONTROL 6
- 2.2 MEASUREMENT DATA MANAGEMENT 7
  - 2.2.1 STRUCTURES 7
  - 2.2.2 WAVEFORMS 8
  - 2.2.3 DATA PROCESSING LISTS 11
  - 2.2.4 DATA FILES 12
- 2.3 PLOTTING WAVEFORMS 13

3.0 INTERACTIVE FUNCTIONS 14
- 3.1 INSTRUMENTATION CONTROL 15
  - 3.1.1 RESET GPIB BUS (Ctl/F6) 15
  - 3.1.2 SELECT (F5) 15
  - 3.1.3 RESET CHANNELS (Shf/F6) 16
  - 3.1.4 ARM CHANNELS (F6) 16
  - 3.1.5 READ CHANNELS (F7) 16
  - 3.1.6 CHANNEL DEFAULTS (Shf/F5) 16
  - 3.1.7 MISCELLANEOUS FUNCTIONS (Alt/F6) 17
- 3.2 MEASUREMENT DATA MANAGEMENT 17
  - 3.2.1 FILE (Ctl/F8) 18
  - 3.2.2 COPY WAVEFORM (F8) 19
  - 3.2.3 COPY X&Y (Shf/F8) 21
  - 3.2.4 COPY Y (Alt/F8) 21
  - 3.2.5 EDIT WAVEFORM (Shf/F7) 21
  - 3.2.6 EDIT PROCESS LIST 23
  - 3.2.7 EDIT RAW ARRAY 24
  - 3.2.8 DEJITTER WAVEFORM (Alt/F7) 25

3.2.9 MEASURE WAVEFORM (Ctl/F7) 25
3.3 WAVEFORM PLOTTING 25
3.3.1 PLOT (F9) 25
3.3.2 PLOTTER (Shf/F9) 27

4.0 PROGRAMMING WITH DAQ 28
4.1 ERROR PROCESSING 28
4.2 STRUCTURES 28
4.3 WAVEFORMS 29
4.4 FILES 31
4.5 WAVEFORM DEJITTER 32
4.6 DIGITAL FILTERS 32
4.7 INSTRUMENTATION CONTROL 34
4.8 PLOTTING GRAPHICS 34

REFERENCES 36

1.0 INTRODUCTION

This technical note describes a data acquisition system designed at DREO for the measurement of very fast transients. It is part of the DREO EMP (electromagnetic pulse) testing facility used for studying the EMP phenomena and verifying the hardening of equipments. These transients are produced by high-voltage generators suddenly discharging into a transmission line. The field generated is of the order of 50 kV per meter, with a rise time of about 5 nsec. This field induces currents and voltages in the object under test. Fields, currents or voltages are measured with the use of appropriate sensors connected to fast digitizers. A personal computer (PC) is used to control the digitizers and to read and process the data.

1.1 INSTRUMENTATION

Several digitizers are used to provide a multi-channel capability. They all feature a very fast single-shot sampling rate and a wide vertical amplifier bandwidth. They are all fully programmable through an IEEE-488 interface and can be operated in stand-alone mode (ie. without a computer). In the current configuration, up to 5 simultaneous channels are available. The table below lists the key characteristics of the instruments (G·s/sec stands for *'Giga-samples per second'*).

Qty	Model	# of channels	Bandwidth	Sampling rate
1	SCD1000	1	1 GHz	200 G·s/sec
2	TEK7912HB	1	700 MHz	100 G·s/sec
1	DSA602	2	400 MHz	2 G·s/sec - 1 channel 1 G·s/sec - 2 channels

1.2 SYSTEM DESCRIPTION

All the digitizers are connected to a computer through the use of a single IEEE-488 bus (also called a HPIB or GPIB bus). Although most of their functions are programmable, most adjustments are made in local mode from the front panel and the computer will only control a few functions.

The computer is a 286- or 386-based personal computer with at least 640 Kbyte of memory and a math co-processor. It is preferable however to use a 386-computer with more than 2 Mbyte of memory. The current implementation uses a 33 MHz 386-computer with a 80387 co-processor, 8 MByte of memory, 150 MByte hard-disk and a VGA adapter and monitor. The system is complemented with a HP LaserJet Series II laser printer and a HP 7550 pen plotter. The laser printer can also emulate a HPGL plotter with a special cartridge (HPGL™ Emulation, from Pacific Data Products).

All the software for the control of the instruments and for the analysis of the data is designed for the ASYST environment. ASYST is an integrated

software designed for instrument interfacing via IEEE-488, RS-232, A/D, D/A and digital I/O interfaces. It provides a full range of scientific analysis and programming capabilities. It can execute commands interactively or can load compiled programs.

The package developed at DREO is called DAQ, for Data AcQuisition. It runs under ASYST. It integrates instrumentation control, data transfer, data management and plotting capabilities.

1.3 SYSTEM CONFIGURATION

ASYST is memory-hungry, particularly in the first 640 KByte (the famous DOS 640K barrier), and thus must be configured carefully to minimized the amount of conventional memory (ie. below 640K) required. It is assumed throughout this document that the reader is familiar with DOS and ASYST.

1.3.1 CONFIGURING DOS

To free-up as much conventional memory as possible and also to provide expanded memory support, a memory manager such as QEMM should be used. It allows one to load resident programs in high memory (ie. between 640K and 1MEG) thus freeing memory. It fully implements the EMS 4 expanded memory specifications. It may also be necessary to increase the number of file buffers and the environment size. A typical 'CONFIG.SYS' file would be:

```
BUFFERS=24,1/X
FILES=30
FCBS=8,2
LASTDRIVE=Z
SHELL=C:\COMMAND.COM /E:600 /P
STACK=0,0
DEVICE=C:\QEMM\QEMM386 RAM options...
DEVICE=C:\QEMM\LOADHI.SYS /B MOUSE.SYS
DEVICE=C:\QEMM\LOADHI.SYS /B C:\DOS\ANSI.SYS
```

where the 3 first lines increase the size of various DOS buffers. Note that the 'FILE=' and 'FCBS=' options can be done with some QEMM utilities in the 'AUTOEXEC.BAT' file, freeing a little more memory. The 'LASTDRIVE=' option must define at least drive Q: (V: if the PC-NFS network is used). The 'SHELL' option increases the size of the environment. The 'STACK=' option removes all stack support (this capability is almost never used by DOS applications). The first 'DEVICE=' option loads the memory manager and subsequent ones are used with 'LOADHI.SYS' to load programs and drivers in high memory.

If a RAM disk (virtual disk) is defined, DAQ will automatically use it for temporary storage and will also copy its own overlays to it for faster execution. An empty file named 'C:\-RAMDISK.d' must be created, where 'd' is the drive letter.

The ASYST environment is initialized during the 'AUTOEXEC.BAT' execution. The following lines should be placed in it:

```
CALL d:\ASYST\SET-UP *
SUBST Q: d:\ASYST
```

where 'd:' is the drive where ASYST resides.

1.3.2 CONFIGURING ASYST

Configuring ASYST consists of loading few system overlays permanently, defining the size of various buffers and arrays, loading a common user library, and saving the result in a file (named 'CNF-LIB'). The procedure described below is done only when installing a new version of ASYST, installing it on a different machine, or when the library is updated.

To configure ASYST:

Change directory to Q:\V3-10 (where the original ASYST resides)

Start ASYST and enter the configuration mode ('<F2>' key)

Specify the hardware configuration:
- 386 processor
- 25 MHz speed
- VGA graphics
- HP LaserJet printer

Load the following overlays permanently:
- Data file
- HP plotter
- GPIB master
- GPIB drivers type-2

Setup the GPIB options (select bus #0):
- Board type-10 (IOtech GP488 board)
- ME address = 0
- I/O address = 2E1
- Interrupt = 7 (not used in this version of DAQ)

Set the memory configuration:

Symbol table size:	24 Kbytes
String segment size:	8 Kbytes
DAS buffer size:	0 Kbytes
GPIB queue size:	0 Kbytes
User dictionary size:	42 Kbytes
Token heap size:	48 Kbytes
Unnamed array (heap) size:	48 Kbytes
Keyboard buffer size:	300 bytes
System buffer size	16384 bytes

Note that the system buffer will be too small, but will be dynamically increased and moved to expanded memory during execution.

Use the command 'LOAD \USERS\LIBRARY\CNF.300' to load the user library.

Enter the memory configuration again and minimized the system (specify file name 'CNF-LIB').

Reload the file 'CNF-LIB' and save with '$SAVE' (specify 'CNF-LIB' for output).

1.4 LOADING THE DAQ SOFTWARE

The DAQ software is located into the '\USERS\DAQ' subdirectory. It also uses some programs found in a library in the subdirectory '\USERS\LIBRARY'. DAQ uses the ASYST concept of applications overlays. These are compiled versions of programs which can be loaded only when necessary. They need to be recompiled only when one of the programs is modified. The loading procedure described below automatically checks if all the overlays are valid.

```
FORGET.ALL
CHDIR \USERS\DAQ
LOAD .\DAQ.
```

It may take few minutes to load all of the programs. It is also possible to save the executable image after loading all of the programs with the '$SAVE' command. When ASYST is loaded by invoking this new file, it is not necessary to use the loading procedure above.

This loading procedure may fail if some invalid overlays are found. The following commands are used to recompile all overlays:

```
FORGET.ALL
CHDIR \USERS\DAQ
CREATE.OVERLAY DFILTERS.MKO
CREATE.OVERLAY FILE.MKO
CREATE.OVERLAY PLOT.MKO
LOAD .\DAQ.1
CREATE.OVERLAY DAQ-EDIT.MKO
CREATE.OVERLAY DAQ-FILE.MKO
CREATE.OVERLAY DAQ-PLOT.MKO
CREATE.OVERLAY SCOPE.MKO
LOAD .\DAQ.2
```

which may be followed by a '$SAVE' command.

1.5 MULTI-TASKING WITH DESQVIEW

ASYST has been successfully run under the DESQview multi-tasking environment. DESQview is a windowing system that allows execution of several applications simultaneously. The distributed version of ASYST is too big to fit

in a DESQview window and the minimized version described in the Section 1.3.2 must be used. It is also important that any memory resident programs or drivers be loaded above 640K; the use of a memory manager such as QEMM386 is thus essential.

2.0 DATA ACQUISITION PACKAGE

This data acquisition software, called DAQ, was developed at DREO for the acquisition and processing of very fast transients encountered during EMP testing. It has the capability to control the instrumentation, to perform data transfer, to execute various digital signal processing algorithms, to maintain a data base of measurements, and to provide on-screen and hardcopy plotting.

Upon loading (as described in Section 1.4), most of the DAQ operation is interactive while some other functions can be invoked in a command line or included in a user program. The interactive functions are accessible through the function keys '<F5>' to '<F10>'. The commands 'DAQ.FKEYS.ON' and 'DAQ.FKEYS.OFF' are used to enable and disable the function keys operation.

Most of the DAQ functions can be divided in three different classes:

- instrumentation control and data transfer
- measurement data management, including digital signal processing
- plotting of measurements, on screen or hardcopy

2.1 INSTRUMENTATION CONTROL

Although all the digitizers listed in Section 1.1 can be fully operated from either their front panel or through a IEEE-488 bus, it was chosen that all of the instrument setup and adjustment (vertical gain and coupling, horizontal sweep control, trigger, etc.) would be done from the front panel. This approach has the obvious advantage of reducing the software required (and a lot of programmer's time), but also it makes it easier to adjust the instrument because of better feedback with the operator. It also keeps the capability to use some of the advanced features available on some instruments (eg. the DSA602 has the capability to perform local data processing).

The only functions which are controlled by the computer, through the IEEE-488 bus, are:

- Reset of the instrument
- Arm the instrument for a single sweep acquisition
- Transfer of data acquired by the instrument to the computer
- Digitize the ground reference (some digitizers only)
- Digitize graticule defects (some digitizers only)

The software controlling the instruments uses the '*device drivers*' concept. With this approach, the procedure to request a particular action or data from an instrument is the same for every instrument and thus the application program need not know about the syntax and protocols particular to each instrument.

In this version of DAQ, the various functions are implemented in:

- File 'SCOPE.ASY' — Standard calls to perform actions; the appropriate device driver is then called.
- Files 'TEK1000', 'TEK7912.ASY' and 'TEK602.ASY' — These are the device drivers for the SCD1000, 7912HB and DSA602 respectively.
- File 'DAQ.ASY' — This is the application program, which in this case maps the various operations to the function keys.

2.2 MEASUREMENT DATA MANAGEMENT

During an experiment, each event (single shot) is recorded on one or more channels, and the experiment may be repeated many times, after possibly modifying some of the setup such as moving to the next test point. After the data has been transferred from the digitizers, it is usually displayed to check the accuracy of the measurement and may also be processed. It is most likely to be stored on disk for later retrieval and analysis. When the data is recorded, it is very important to retain information which will describe as fully as possible the setup and equipment used during the experiment. In addition to the raw data (a sequence of points, in volts, at equally spaced intervals), additional information should include: the time step and possible a time offset, the vertical scale and offset, the date and time of the experiment, a title, a description of the setup (eg. sensor and cable used). It is important that all the information relevant to a measurement done on one channel remains together when manipulated or stored on disk. To address that problem, two concepts were introduced: '*structures*' and '*waveforms*', which will be discussed in more detail later in this section. Structures allow multiple objects to be packed into a single array. Waveforms define how a structure is partitioned to store all the information relevant to a measurement.

2.2.1 STRUCTURES

A structure is defined as a collection of one or more variables, possibly of different types and varying size, grouped together in a single array. This concept is similar to what many high-level computer languages define as structures or records. There is however one major difference: the type and size of a variables to be stored in a structure is defined dynamically.

Structures are implemented using single precision integer arrays. They are created with an user-specified number of items, which are initially empty. Scalar, single-dimensioned arrays or character strings can be stored in an item. Numeric values may be single precision integers or single precision reals (double precision or complex values are not currently allowed). Items can be extracted or replaced; new items can be added and some may be deleted.

The internal definition of a structure, shown below, contains a header, a list of pointers (one set of pointers for each item) and the content of each item. Each pair of brackets ("[ ]") represent one single precision word.

```
                                    Header area:
["{}"]                                 2-byte string (value 32123), identi-
                                       fying the array as a structure

[user-id]                              available to the user to identify
                                       the content of the structure

[user-option]                          available to the user

[# of items]                           # of items in the structure

                                    Pointers area (one set for each item):
[beg]  [end]  [type]                   'beg' and 'end' are index within
 ...    ...    ...                        this array pointing the data area
                                           of this item
                                       'type' refers to the data stored:
                                           1=Integer, 3=Real, 9=String
                                       All zero if item is empty.

                                    Data area:
[data] [data] ... [data]               data of 1st non-empty item
[data] [data] [data] ... [data]        data of next non-empty item
 ...    ...    ...   ...
```

The header and pointers area are always present while the data area is dynamically adjusted as data is stored or deleted. All data is converted to single precision integers before being stored and are reconstructed when it is extracted. Single precision reals are mapped as two integers (4 bytes) and strings are packed as two characters in an integer. It should be noted that because a structure varies .dynamically in length, it cannot be stored in a named-array, and thus tokens must be used. Tokens have the additional advantage of using expanded memory for storage.

2.2.2 WAVEFORMS

A waveform is a structure containing all the information pertinent to one measurement. The information is obtained from the digitizers, the setup, the user, or is generated by DAQ. A example of a waveform is shown here:

```
  Comment:▸This is a sample waveform
Tr. Desc: L1 on MAIN          Time step: 1.9570E-10      Ch: 4    V.scale
   Label: 377 Ohm           Time offset: 0.0000E-1     Shot: 0    V.offset
    Date: 91/08/29 16:03:37 User offset: 2.0033E-2   Sensor:      0  11944
    Data: 512 pts              Att.(dB): 3.0000E1     Cable:     0      0
 Process: 1 items          Sensor scale: 2.2599E11    Scope:  1000  16089
```

where the source and description of each field is:

from digitizers:	
time step (or ΔT)	Time in second between data points.
vertical scale and offset	Raw data is (y+offset)·scale.
raw data	An array of up to 2048 data points (integer or real).
trace description	Some digitizers such as the DSA602 have the option of defining traces.
from the setup	
DAQ channel #	8 channels are defined, 6 are currently used.
digitizer on this channel	Digitizer used: model # and id. (or asset #).
cable used	Cable used: model # and id.
sensor used	Sensor used: model # and id. (or asset #).
sensor scaling factor	Factor to convert volts to field intensities or other quantities.
attenuation	Sum of all attenuators used.
from the user	
comment or title	To identify the experiment.
label	Short title to identify a particular option. Will be included in hardcopy plots.
shot #	A # given to this measurement.
time offset	For aligning multiple waveforms (such as a trigger offset).
user offset	An additional offset (in Volt) added to the raw data.
data processing	List of operations (signal processing) to performed on the raw data.
from DAQ	
date and time	Date and time of the data acquisition.

The information from the setup and the user may be entered into a 'default waveform', which is automatically used when creating a new waveform when reading the data from the digitizers.

All the fields defined above are amalgamated together into a single array as a structure. The internal definition of a waveform is shown below. Brackets ("[ ]") denote indexing of an array.

	Structure header:
[1]	String "{}" (value 32123)
[2]	String "WF" (value 18007)
[3]	=102, identifies version 1.02
[4]	=7, # of items in waveform.

Item 1	Character string. The comment, trace description and label field are packed into it.
Item 2	Integer parameters (must all be present)
[1]	Channel #
[2] & [3]	Sensor type and id.
[4] & [5]	Cable type and id.
[6] & [7]	Digitizer type and id. (asset #)
[8]	Shot #
Item 3	Real parameters (must all be present)
[1]	Time step (ΔT), origin 0.
[2]	Vertical scale factor (0 if none)
[3]	Vertical offset (0 if none)
[4]	Time offset (0 if none)
[5]	Sensor scaling factor (0 if none)
[6]	Attenuation, in dB (0 if none)
[7]	User offset
Item 4	Date & time, packed into a DIM[3] array
Item 5	May contain a structure representing a list of data processing operations to be performed.
Item 6	Raw data, usually an integer array. When extracted, it is modified by vertical scale and offset.
Item 7	Processed data. This item stores the resulting processed data array, it is not usually stored in files as it can be easily calculated again.

Only items 2 and 3 must be present at all times. Other items are created only when necessary.

The version number identifies the version used when creating a waveform. Whenever a waveform recalled from file is from a previous version, it is automatically converted to the current (latest) version.

Item 1 is the concatenation of all fields defined as strings. It has the following format:

"main_title_field\TRA:trace_description_field\LAB:label_field"

where the main title, if present, is first, and additional fields, if any, follow (in no particular order) using the syntax "\XXX:string" where 'XXX:' represent a user-defined header.

When the raw data is extracted from a waveform, it is modified only by the vertical scale and offset, yielding to a quantity in Volt:

$$\{y_{raw}\} = (\{x\}+V_{offset})\cdot V_{scale}$$

where {x} is the array (usually an integer array) transferred from the digitizer. Similarly, the processed data array is calculated as:

$$\{y_p\} = (\{y_{raw}\} + User_{offset}) * Sensor_{scale} * 10^{Attn/20} \quad \ldots$$

followed by the operations listed in the data processing list.

2.2.3 DATA PROCESSING LISTS

The data processing to be performed on a waveform is also stored along with it in item 5. If present, this field contains a structure representing a list of operations to perform, one operation per item. Each item consists of an integer or real array (possibly only one element), where the first element is a code representing the operation. The format for the items of a processing list is shown below. Brackets ("[ ]") denote indexing of the item.

[1]=01		Scale by [2]
=02		Offset by [2]
=03		Scale by 1/[2]
=08		Integrate (use ΔT). Also scale by [2] if present.
=09		Integrate offset for end value=[2]
=10		General FIR filter.
	[2]	Valid for ΔT=[2] (0=valid for all ΔT)
	[3..n]	Coefficients
=11		General FIR filter.
	[2..n]	Coefficients are stored in file. File name packed in [2..n]
=20		General IIR filter — h(z)=b(z)/a(z)
	[2]	Valid for ΔT=[2] (0=valid for all ΔT)
	[3]	size of a(z)
	[4..n]	Coefficients a(z)
	[n+1..m]	Coefficients b(z)
=21		General IIR filter.
	[2..n]	Coefficients are stored in file. File name packed in [2..n]

-22 [2] [3..]	General IIR filter, cascaded form. Valid for ΔT=[2] (0=valid for all ΔT) Coefficients (Nx6 array) stored in a vector
-31	1st order low-pass filter, $H(s)=1/(s/\omega_c+1)$, f_c (in Hz) stored in [2]
-32	Inverse of a 1st order hi-pass filter, $H(s)=(s/\omega_c+1)/(s/\omega_c)$, f_c (in Hz) stored in [2]
-33	Filter performing a partial integrator recovery (see reference [1]), $H(s)=(s/\omega_c+1)/s$, f_c (in Hz) stored in [2]
-35	N^{th}-order Butterworth low-pass filter, N stored in [1] and f_c (in Hz) stored in [3]
-41	1st order hi-pass filter, $H(s)=(s/\omega_c)/(s/\omega_c+1)$, f_c (in Hz) stored in [2]
-42	Inverse of a 1st low-pass filter, $H(s)=s/\omega_c+1$, f_c (in Hz) stored in [2]
-45	N^{th}-order Butterworth hi-pass filter, N stored in [1] and f_c (in Hz) stored in [3]

A process item can be disabled, but kept in the list, by storing the negative of the code in [1].

Conversion from the Laplace transform to the z-transform is performed by using the bi-linear transformation (Reference [2]):

$$s = \frac{2}{\Delta T} \frac{1-z^{-1}}{1+z^{-1}}$$

2.2.4 DATA FILES

DAQ makes use of data files, as defined by ASYST. They consist of a number of comment lines followed by numerical data stored in 'subfiles'. The main limitation with ASYST data file system is that once a subfile is defined, its size and structure cannot be changed. As the data stored in waveforms vary in size, a new file structure, using ASYST data files, was defined. It is somewhat similar to the structure of a disk: subfiles, like blocks on a disk, all have the same size. The data is stored into one or more subfiles (blocks), not necessarily consecutive, and pointers are use to maintain a linked-list.

DAQ data files can store up to 100 waveforms which can be randomly accessed, ie. numbered from 1 to 100. A block size of 600 is used (the number of data points from digitizers is often a multiple of 512, and adding the additional information to store in a waveform makes them usually a little less than a multiple of 600). However, DAQ will correctly read data files with

different block sizes. The structure of DAQ data files is as follow: 4 comment lines, 3 index arrays, and 1 to 999 data blocks; all arrays are single precision integers:

Comment#	
1	Header = "(DAQ) VARRAY - Struct & wfm on disk"
2	reserved
3	Title or user comment
4	Available to user

Subfile#	
1	(size) DIM[100] array, size of the waveform stored.
2	(lst) DIM[100] array, pointer to the first data block.
3	(idx) DIM[999] array, pointer to the next data block. Set to -1 if this is the last block of data. Set to 0 if the block is unused.
4-1002	Data area. Data block #1 correspond to subfile #4.

This structure will accommodate files as large as 1 MBytes (about 500,000 data points). However DAQ can be easily reconfigured to accommodate larger files (see file 'DAQ-VARR.ASY').

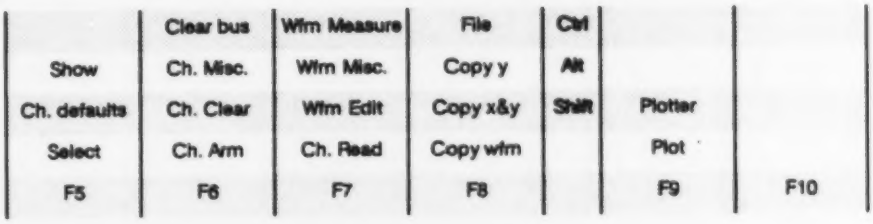
2.3 PLOTTING WAVEFORMS

DAQ has the capability of plotting up to 20 waveforms simultaneously. Every time a waveform is added to the plot, some information (x and y arrays, minimum and maximum values, and the label field) is stored in a trace file ('DAQ{PLT}.TMP') which is used to either refresh the plot on screen or to create a hardcopy plot. Originally, hardcopy plots were made on a HP pen plotter such as a HP 7550. It is now possible to use a HP LaserJet to produce black & white plots. This is possible by using a special cartridge (HPGL™ Emulation, from Pacific Data Products) which emulates the HPGL plotter language. It is also possible to send a hardcopy plot to one or more files. These files can be imported directly in WordPerfect or DrawPerfect to include in reports.

DAQ does not use the ASYST primitive functions 'XY.AUTO.PLOT' and 'XY.DATA.PLOT' but instead it uses the new functions 'XY.PLOT', 'XY.DPLOT' and few others (found in the file 'PLOT.LIB') which produce better results. Other functions found in 'PLOTTER.UTL' implement plotter redirection and utilities. Refer to Section 4.8 for a description of the plotting routines

3.0 INTERACTIVE FUNCTIONS

The interactive functions are accessible through the function keys '<F5>' to '<F10>'. The function keys '<F1>' to '<F4>' retain their definition as defined by the 'LINE.EDIT' command. The function keys '<F11>' and '<F12>' are reserved for future use. The commands 'DAQ.FKEYS.ON' and 'DAQ.FKEYS.OFF' are used to enable and disable the function keys operation. The template below shows the assignment of each key:



'DAQ.FKEYS.ON' will in turn call 'DAQ.INIT', which will initialize all arrays and parameters if not done already. A GPIB bus reset will also be performed. Either 'DAQ.FKEYS.ON' or 'DAQ.INIT' must be called at least once before using any of the DAQ functions ('DAQ.INIT' could be called if the interactive functions are not necessary).

If a RAM disk is available (see Section 1.3.1) the commands 'RAM.DISK.ON' and 'RAM.DISK.OFF' can be used to enable or disable its use by DAQ. When used, the DAQ application overlays and all temporary files are copied to it and executed from there.

Most functions, when activated by pressing one of the function keys, also prompt the user for additional input. This is in the form of a single letter, a number, a list of numbers, or a string. All inputs, except for the single letter input, are terminated with '<CR>'. When the user is asked for a single letter input, the prompt highlights the possible choices for the answer. If '#' is one of the choices, a single digit number may also be typed (the allowed range is defined by the context). In some cases, some hidden choices are available; they are usually shortcuts to specify an argument. In general, if '_channel' is one of the choices, typing a single digit (1-8) will select that channel. Pressing '<CR>' only usually specifies a default action. In this report, the highlight is shown as a underline and the default action as a double underline. At any time, the '<ESC>' key can be used to cancel the current operation, returning to the ASYST prompt or one of the previous prompts.

DAQ assumes that it is located on drive Q: (see Section 1.3.1 for proper configuration) in the directories 'Q:\USERS\DAQ' and 'Q:\USERS\LIBRARY'. Temporary files are also stored initially in 'Q:\USERS\DAQ'. To display the current location of various items, the 'SHOW logicals' function, invoked with the 'Alt/F5' key produce the following listing (without the comments on the right):

DAQ:	Q:\USERS\DAQ\	The DAQ program files
LIB:	Q:\USERS\LIBRARY\	The library files
SOV:	E:\ASYST\V3-10\	The ASYST program and overlays
AOV:	Q:\USERS\DAQ\	The DAQ overlays
TMP:	Q:\USERS\DAQ\	Temporary files

3.1 INSTRUMENTATION CONTROL

There are in the present configuration 6 instrumentation channels (or traces) defined. Note that this correspond to only 5 real inputs to the digitizers; instruments like the DSA602 allow to define more traces than the actual number of inputs.

The 'SHOW GPIB' function, invoked with the 'Alt/F5' key produces the following listing:

Ch #	Type	#tr	GPIB ad.
1	TEK7912	1	3
2	TEK7912	1	4
3	SCD1000	1	6
4-6	DSA602	3	5

which shows the model of digitizer, the number of channels (traces) for each, and the GPIB address.

The template below shows which function keys are used to control the instruments:

	Clear bus	Wfm Measure	File	Ctrl		
Show	Ch. Misc.	Wfm Misc.	Copy y	Alt		
Ch. defaults	Ch. Clear	Wfm Edit	Copy x&y	Shift	Plotter	
Select	Ch. Arm	Ch. Read	Copy wfm		Plot	
F5	F6	F7	F8		F9	F10

3.1.1 RESET GPIB BUS (Ctl/F6)

This function, which does not take any argument, performs a reset of the GPIB bus. It is automatically called by the 'DAQ.FKEYS.ON' and 'DAQ.INIT' functions.

3.1.2 SELECT (F5)

All the functions interfacing with the instrumentation can be performed on a single channel or on multiple channels. When an operation applies to multiple channels, it uses a previously stored selection. This command is used to store the selection. At the "Select channels: " prompt, simply enter the selection, each entry separated by a space.

The 'CLEAR', 'ARM', 'READ' and 'MISC. FUNCTIONS' functions prompt the user to select single or multiple channels for the operation. The prompt for the channel selection is always in the form:

"...select single or all selected channels: "

where answering 'single' brings the additional prompt "channel #: " and typing 'all' or '<CR>' recalls the previously stored selection. It is also possible to use a shortcut to specify a single channel by typing a single digit corresponding to the channel # (this is a hidden choice).

3.1.3 RESET CHANNELS (Shf/F6)

This function resets one or multiple digitizers. It only prompts for single or multiple channels as described in the previous section. A short status is displayed as the operation proceeds through the selection list. If two or more channels specify the same instrument, the reset operation will be performed only once; the status 'skip' will be displayed. The possible status codes are:

Ok	Success
tmo	Timeout — it usually means that the instrument is offline.
skip	Operation already performed
Err ###	Other error, an error code follows

If '<ESC>' is pressed while the command is executing, the operation will be aborted before going on with the next channel.

3.1.4 ARM CHANNELS (F6)

This function arms one or multiple digitizers for a single sweep data acquisition. It assumes that the instrument is ready for single sweep (proper horizontal sweep speed, proper vertical coupling and gain, proper trigger, etc.). Its prompt and status reporting is similar to the 'RESET' function.

3.1.5 READ CHANNELS (F7)

This function is used after the digitizers have been armed and have recorded an event to transfer the data into newly created waveforms. The waveforms are stored into the arrays (tokens) CH1 to CH8. The waveforms follow the format described in Section 2.2.2, which incorporate data from the digitizers and from the setup (see next section). Its prompt and status reporting is similar to the 'RESET' function.

3.1.6 CHANNEL DEFAULTS (Shf/F5)

As described in Section 2.2.2, a waveform stores the data obtained from a digitizer along with some information regarding the setup used. To avoid to have to fill-in the setup information for every measurement, a default waveform (like

a template) is used for each channel. This function allows one to create, edit and delete the default waveforms, one for each channel. When editing a default waveform, the waveform editor (described in Section 3.2.5) is called. Only the following fields (highlighted by the editor) can be specified in a default waveform: the title, the label, the time offset, the user offset, the cable type and id., the sensor type and id., the sensor scaling factor, the attenuation, and the process list.

Additionally, this function allows one to save all the default waveforms in a file, and later retrieve them. The user is prompted for a file name and the default name 'DAQ(DEF).CHD' is used.

3.1.7 MISCELLANEOUS FUNCTIONS (Alt/F6)

This function key implements miscellaneous functions particular to some digitizers. The main prompt selects one of the only two functions implemented:

"Digitize gnd ref(clear) or defects: "

The 'Digitize ground reference' function is particular to the TEK7912 and DSA602 digitizers. It is used to measure the zero Volt reference (input shorted). The 'Clear ground reference' function is used to clear a previously stored ground reference. The 'Digitize defects' is also particular to the TEK7912. It is used to locate and record the defects on the internal graticule. It should always be used after the instrument has warmed-up, and before any accurate measurement are made.

The user is then prompted for single or multiple channels. If 'multiple channels' is selected, the following prompt allow to further restrict the operation to only one type of digitizers if desired:

"For 1. TEK7912, 2. SCD1000, 3. DSA602 or all: "

The status reporting is similar to the 'RESET' function.

3.2 MEASUREMENT DATA MANAGEMENT

This group of functions implements a waveform data base system, allowing one to manage measurement data (waveforms) in memory and on disk. It allows one to store the acquired waveforms on disk and to recall them for plotting or for further processing. It also allows one to modify (edit) a waveform interactively.

In general, DAQ offers several options for the source or destination when manipulating or copying a waveform. These options may be:

channel	Waveforms obtained with the 'CH. READ' function. They are stored in the variables (tokens) CH1 to CH8.

file	User file, in the format described in Section 2.2.4. Each file may contain up to 100 waveforms. It is also possible to create an ASCII file containing the x & y arrays of a waveform in a tabular form.
workspace	File used for temporary storage of waveforms ('DAQ{WKS}.DAT').
stack	From or to the current stack. The stack may also be the destination when the x and y arrays of a waveform are extracted.

For a given operation, all the options for the source or the destination may not be available. The general prompt to specify a source is:

"from channel, stack, file or workspace: "

where single digits 1 to 8 can also be used as a shortcut to specify a channel. Similarly, the general prompt for a destination is:

"to stack, file or workspace: "

To avoid typing the file name every time a file is accessed, up to 4 names may be entered and files are simply numbered 1 to 4. If 'file' is selected, the user is prompted for a file number. If none is specified ('<CR>'), the last number specified during a file operation is used. Then, for both 'file' and 'workspace', the user is prompted for an item #, in the range of 1 to 100. If none is specified during a write operation, the first available (empty) item is chosen. The item # must be specified for read operations.

The template below shows which function keys are used to implement the waveforms management functions:

	Clear bus	Wfm Measure	File	Ctrl		
Show	Ch. Misc.	Wfm Misc.	Copy y	Alt		
Ch. defaults	Ch. Clear	Wfm Edit	Copy x&y	Shift	Plotter	
Select	Ch. Arm	Ch. Read	Copy wfm		Plot	
F5	F6	F7	F8		F9	F10

3.2.1 FILE (Ctl/F8)

This function provides general file management. The main prompt selects one of the three functions implemented:

"Name, create or list file: "

The 'Name file' function is used to specify up to 4 different file names and also to specify a default file name specification. This function can also be accessed directly from the main prompt with 'd' to change the default or with

the digits 1 to 4 to change one of the names (these are hidden choices). Whenever a new file is created or a new name is specified, the default '.*.DAT' is applied, then the default name is used to establish a default path, name and extension.

Files must exist before they can be used as a source or a destination. The 'Create file' function allows one to create a new workspace file or a new data file. If a new data file is being created, the user is asked a file # and file name. The file name replaces the previous name. The user can also specify a title which will be saved in the file.

The 'List file' function list the content of a file on the screen or in a file. The user is prompted for the name of the file to list:

"List workspace or file #: "

where '#' is a file number (1-4) as described above. If no file is specified ('<CR>'), the last file number specified during a file operation is used. If 'file' is selected, the user is prompted for a file name (which will not be stored for further use). The user is also prompted for the destination of the listing:

"List to screen or file(long): "

where the default is on screen. If the listing is sent to a file, its name will be the same with the '.LIS' extension in the current directory. Each item present in the file is shown, with its title, label, date of acquisition, channel # and time step. A typical file listing is shown here:

```
..\DAQ\EXPERMNT.DAT  Measurement of performance of partial integrator..
# Date           Ch Time step  Label          Comment
1 91/08/29 14:34  3 1.9570E-10 Volt.probe      Voltage probe
2 91/08/29 14:40  3  1.9569E-9 Volt.probe      Voltage probe
3 91/08/29 14:47  3 1.9570E-10 d-dot           d-dot, @ 1m
4 91/08/29 15:21  3 1.9570E-10 d-dot+PI @scop d-dot, @ 1m, Part.Int.
5 91/08/29 16:03  3 1.9570E-10 d-dot+PI @scop d-dot, @ 1m, Part.Int.
6 91/08/29 15:58  3  1.9569E-9 d-dot+PI @scop d-dot, @ 1m, Part.Int.
```

where the display (except the title) will scroll. Note that main title and some fields are truncated to fit more information on a single line. While a file is being listed, any key can be typed to suspend the output to the screen. '<CR>' can then be typed to advance one line at the time, or any other key to resume normal listing. The '<ESC>' key can be used at any time to abort the listing.

If the 'long' option is chosen, a complete listing of every waveforms is produces, as shown on Figure 1.

3.2.2 COPY WAVEFORM (F8)

This function simply copies a waveform from the specified source (channel, stack, file or workspace) to the specified destination (stack, file or workspace).

```
*******************************************************************************

  File: Q:\USERS\DAQ\EXPERMNT.DAT
 Title: Measurement of performance of partial integrator

*******************************************************************************

 Comment: d-dot, @ 1m, Part.Int. @scope
   Trace:
   Label: d-dot+PI @scope

          Item #: 4                    Date: 91/08/29 15:21:24
            Ch #: 3                Raw data: 512 pts
          Shot #: 0

       Time step: 1.9570E-10         Scope: 1000 (TEK SCD1000)  Id: 16089
     Time offset: 0.0000E-1          Cable: 0                   Id: 0
  Vertical scale: 4.8852E-4          Sensor: 0                  Id: 11944
Vertical offset: -4.0990E2                                   Scale: 2.2599E11
     User offset: 2.3000E-2
     Attenuation: 3.0000E1 dB

      Processing:   Partial integrator recovery, (s'+1)/s, fc= 3.0000E7

*******************************************************************************
```

Figure 1. Sample of a full listing of a file.

3.2.3 COPY X&Y (Shf/F8)

This function extracts from a waveform the x & y arrays and places them on the stack (two arrays) or in an ASCII file. The user is asked if raw data (default) or processed data is desired.

If an ASCII file is specified for the destination, the arrays are written to it in a tabular form (two columns). That file can be used to import the data into another program, such as Graftool, for advanced plotting or further processing. This file format should not be used for general storage of waveforms as all other information stored in a waveform is not copied. ASCII files are also must slower to access.

3.2.4 COPY Y (Alt/F8)

This function extracts from a waveform the y array (raw or processed) and places it on the stack.

3.2.5 EDIT WAVEFORM (Shf/F7)

This function key invokes the waveform editor. It can be used to modify any waveform just acquired (channels stored in tokens), on stack or in files. It is also used to specify the default waveforms (see Section 3.1.6); the only difference is that default waveforms limits the editing to only some of the fields. Upon entry, the waveform editor clears a window of 9 lines; the top 6 lines are used to display the waveform and the two bottom lines are used for prompts and messages. The user is initially prompted to select a waveform (from channel, stack, file or workspace):

"Edit/delete from channel, stack, file, workspace or done: "

If 'delete' is chosen, the user is prompted again to select a waveform to delete. After a waveform has been retrieved, the display looks like:

```
  Comment:▸This is a sample waveform
Tr. Desc: L1 on MAIN          Time step: 1.9570E-10      Ch: 4    V.scale
   Label: 377 Ohm            Time offset: 0.0000E-1     Shot: 0    V.offset
    Date: 91/08/29 16:03:37  User offset: 2.0033E-2   Sensor:     0  11944
    Data: 512 pts              Att.(dB): 3.0000E1     Cable:      0      0
 Process: 1 items           Sensor scale: 2.2599E11    Scope:   1000  16089

  Edit/delete from channel, stack, file, workspace or done:        Channel 4
```

where a red cursor, shown next to 'Comment:' is used to select a particular field. The arrow keys are used to move the cursor. Note that the value or content of some of the fields is not shown (like the vertical scale and offset), but they can still be selected for editing. The following keys perform specific functions:

E or {Ins}	Edit this field. The user will be prompted for additional input.
D, {Del} or {BS}	Delete this field.
Ctl/W	Refresh the screen.
{End}	Save the waveform as modified. Return to the main prompt.
{ESC}	Discard all changes made to this waveform. Return to the main prompt.

Some of the fields of a waveform are not usually set by the user. If the user attempts to delete or modify one of these fields, the prompt "Are you sure? " is asked before proceeding. The table below summarizes how every fields are edited . '(P)' refers to protected fields, requiring confirmation before being modified.

Comment	ASCII string. No particular restriction on the length, but it may be truncated for display.
Trace descr. (P)	ASCII string. Truncated to 17 characters for display.
Label	ASCII string. Truncated to 17 characters for display and to 21 for hardcopy plots.
Date (P)	Prompt to change: year, month, day, hour, minutes and seconds. '<CR>' leaves the item unchanged.
Data (P)	'{Del}' will delete the whole array. 'Edit' will bring the additional prompt. The 'Remove defects' option allows the removal of defects on data obtained for a TEK7912 digitizers. This is only an estimation of what the correct data should be. The 'Digitize defects' ('Alt/F6') should be used instead. The 'index array' option allows one to keep only a subarray of the original by specifying the first and last indices. The 'edit array' option calls an array editor (described in Section 3.2.7) that allows to edit each element individually.
Process	'{Del}' will delete the entire list. 'Edit' will invoke the process list editor (described in Section 3.2.6).
Time step (P)	Real number. Should never be set to 0.
Time offset	Real number.
Vertical scale (P)	Real number.
Vertical offset (P)	Real number.

User offset	Real number. Any value can be specified with the 'Set' option, or an estimate can be calculated from the portion of data prior the fast rising edge with the 'Compute' option.
Sensor scale	Real number. Ignored if 0.
Channel # (P)	Integer.
Shot #	Integer.
Sensor type & id.	Integers.
Cable type & id.	Integers.
Scope type & id. (P)	Integers. Usually obtained from the GPIB configuration.
Attenuation	Real number. In dB. Negative numbers are interpreted as a gain.

3.2.6 EDIT PROCESS LIST

When the user requests to edit the processing list (as part of the waveform editor), a different editor is called. It will initially display the first 6 processing items. A typical display looks like:

```
  1:▸ Offset by 1.0000E-1
  2:  Scale by 1.2000E1
D 3:  Integrate
  4:  Low-pass filter, fc= 3.0000E7
  5:  FIR filter, file: cable.fir
   :

 Edit, Insert, Toggle, keypad to select, ? for help.
```

where each item is sequentially numbered and an empty item (not numbered) marks the end of the list. A red cursor, shown next to '1:', is used to select a particular item. Up and down arrows, and the page up and page down keys if there are more than six items, are used to move the cursor. The following keys perform specific functions:

{PgUp} and {PgDn}	Display the previous or next six items in the list. Use with the up and down arrows to select an item.
I or {Ins}	Create a new item. The user is prompted for the type of processing (scale, offset, integrate, low/hi-pass, FIR/IIR, Laplace transform) and for parameters.
E	Edit this item. The user will be prompted for additional input.
D, {Del} or {BS}	Delete this item.

T or -	Toggle the enable/disable flag. Disabled items are shown with the flag 'D' on the left.
Ctl/W	Refresh the screen.
{End}	Save the process list as modified into the waveform. Return to the waveform editor.
{ESC}	Discard all changes made to this waveform. Return to the waveform editor.

The 'Insert' option brings the additional prompt:

"Scale(/), Offset, Integrate(-), Low/Hi-pass, FIR/IIR, Laplace: "

where the 'Scale', '/' and 'Offset' options multiply, divide or add a constant to the waveform, the 'Integrate' and '-' options perform the integrate waveform or integrate offset for end-value operations, the 'FIR' and 'IIR' options allow specification of a coefficient file, the 'Low' and 'Hi-pass' options further prompt the user for the parameters of a low-pass or high-pass filter, and the 'Laplace' option allows selection of one of the special Laplace transfer functions. Note that some forms of processing discussed in Section 2.2.3 require computation and storage of the coefficients (processing codes 10, 20 and 22). These items will be properly displayed, but cannot be edited.

3.2.7 EDIT RAW ARRAY

The array editor, used by the waveform editor to edit the raw data field, displays the content of an array (one page at a time) and allows the modification any element of the array. Upon entry, the display looks like:

1:	385	386	384	379	372
6:	366	358	354	353	355
11:	359	364	368	373	378
16:	383	388	390	388	385
21:	381	376	373	369	364

Edit, (^)PgUp/(^)PgDn, {End}/{Esc}.

where 25 elements are displayed at once, the first column showing the index of the first element on that line. The following keys perform specific functions:

E or {Ins}	Edit. The user will be prompted for an index and a new value for that element.
{PgDn}	Display the next 25 elements of the array. Ctl/{PgDn} displays the end of the array.
{PgUp}	Display the previous 25 elements of the array. Ctl/{PgUp} displays the beginning of the array.
{End}	Save the waveform as modified. Return to the main prompt.

{ESC} Discard all changes made to this waveform. Return to the main prompt.

3.2.8 DEJITTER WAVEFORM (Alt/F7)

The only miscellaneous operation currently implemented is a waveform dejitter function. This function allows compensation of the trigger jitter by realigning one or more waveforms against a reference waveform. This is especially valuable for comparing different waveforms or for signal averaging. The dejitter function uses a least-square algorithm to find the best match between two arrays. The main prompt is:

"Dejitter compute or set/reset reference: "

which is followed with a prompt to specify a waveform if 'compute' or 'set reference' is selected. The 'Set reference' is used to specify a reference waveform (raw or processed array), along with the option to specify a N^{th}-order Butterworth low-pass pre-filter. The 'compute' option computes the time offset necessary to obtain the best alignment between the two waveforms. The result is stored back by adjusting the time offset field of the waveform. The 'Reset reference' option clears some internal arrays to release some memory space.

3.2.9 MEASURE WAVEFORM (Ctl/F7)

This utility simply returns some information about the raw or processed data of a waveform, mainly the positive and negative peaks, and the rise time.

3.3 WAVEFORM PLOTTING

DAQ has the capability of plotting up to 20 waveforms on a single graph. It also has the capability of producing hardcopies on a pen plotter (in color), on a LaserJet printer (monochrome), or in a file (HPGL format). A graph must be produced on screen before a hardcopy can be made. Figure 2 shows a typical hardcopy plot. It includes the plot itself (with or without a grid), a main title, a label for both axis, the date or a range of dates at which the measurements were made, and a list of labels identifying each curves. Graphs displayed on screen do not show this additional information.

3.3.1 PLOT (F9)

This key controls the plot generation. To maintain ASYST terminology, it has two modes: 'Plot' for creating a new graph and 'Data plot' for adding a curve on a graph. Upon entry, the 'Plot' mode is in effect and the keys 'D' and 'P' are used to change mode. The main prompt is:

"D/Plot, from channel/stack/file/workspace, refresh/hardcopy/zoom: "

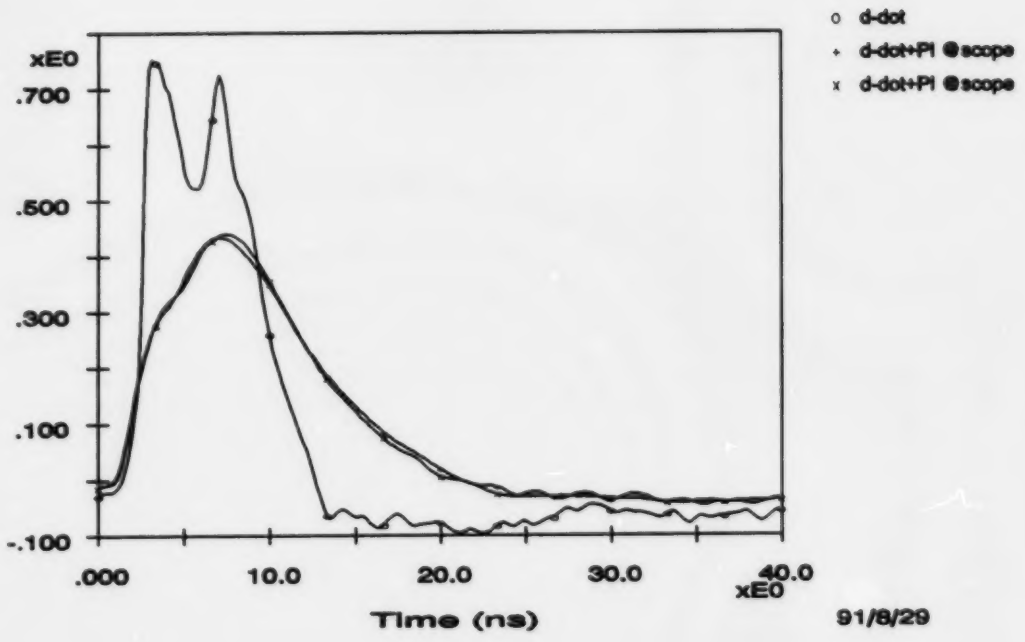


Figure 2. Sample hardcopy plot.

where adding a curve is achieved by simply specifying a waveform (from channel, stack, file or workspace). The user is also prompted for raw (default) or processed data. When adding a new curve on a graph, it is clipped to fit the current borders. The 'refresh' option is used to redraw the graph, with all curves fitting completely within its borders (unless the zoom feature is active). The 'zoom' option allows one to skip the autoscale feature and let the user specify a range for both X and Y axis. If only one value is given for range, then a lower bound of 0 is taken. The 'hardcopy' option produces a hardcopy of the last plot. '<CR>' is use to exit to the ASYST prompt.

3.3.2 PLOTTER (Shf/F9)

This key is used to specify the destination of the hardcopy plots, the number of pens to use, and the title and labels to include in the hardcopy. The main prompt is:

"Plotter destination, # of pens, labels: "

and the prompt to specify the destination is:

"Send plot to COM1, LPT1, spool, file or none: "

If COM1 or LPT1 is specified, it is assume that the device will correctly interpret HPGL commands. If the device is a HP LaserJet, it must be switched to HPGL Emulation mode before any commands are sent. If 'spool' is specified, a temporary file will be created and sent to the device with the DOS PRINT command. This effectively sends the data in background and improves the interactive response of the computer. The 'spool' option however does not work very well with DESQview. It may be necessary to redirect the output of the PRINT command. Consult the DOS manual for additional information on the PRINT command. If 'file' is specified, the user is prompted for a file name. If the name contains an extension, a single file will be created for all the plots (separated with a 'new page' command). If the name does not have an extension, a new file will be created for every plot, with an extension ###, which is a sequential number from 001 to 999.

The '# of pens' option defines the number of colors to use. By convention, pen #1 is always black and is used to draw the box, axis and text; pen #2 is always yellow and is used to draw the grid only. Other pens will be used to draw the curves. For LaserJet outputs, the number of pens should be 1 if the grid is not desired, or 2 if it is desire. If only 1 or 2 pens are used, DAQ will automatically label each curve with a different symbol, as shown on Figure 2.

The 'label' option is used to specify a main title and the x-axis and y-axis labels to be included in hardcopies. If the x-axis label is not specified, "Time (s)" will be used (the unit in parentheses will automatically be adjusted to reflect the range of x-values).

4.0 PROGRAMMING WITH DAQ

To date, the DAQ software contains more than 10000 lines of code, in about 80 files found in the '\USERS\DAQ' and '\USERS\LIBRARY' directories. This chapter, which is definitely not an exhaustive review of all the programs, describes the more important and useful programs to complement the interactive functions described in the previous chapter with the capability of writing user-written programs. The source files are well documented and more information can be found there. The syntax of the various programs given herein is in the format used in the ASYST documentation.

Some of the programs defined in this chapter are precompiled and stored as application overlays ('*.AOV' files). Before these programs can be called, it is necessary to load the overlay, which is done by inserting this line within the program definition:

```
[USE.AOV.PATH]  [COMPILE] [USE.AOV.PATH]
```

Similarly, this line allows the use of an ASYST overlay which is not permanently loaded:

```
[USE.SOV.PATH]  [COMPILE] [USE.SOV.PATH]
```

It should be noted that only one transient overlay can be loaded at a given time; the previous one is automatically unloaded.

4.1 ERROR PROCESSING

Error recovery in ASYST is achieved with the use of a 'ONERR:' clause at the end of a program definition. When an error occurs, the program is aborted and the code following the 'ONERR:' clause is executed. The program may test for error code and decide to handle the error or to propagate it to the caller (which gets an error interrupt). Many programs described in this chapter will perform general checks on arguments and attempt to recover from some expected error conditions. In those cases, the caller will be warned of an error by some means (flag, specific value returned, etc.), and the '?ERROR#' flag and '?ERROR"' string can be used to determine the exact cause of the error. Unexpected errors are generally simply propagated and will probably result in the main program being aborted.

4.2 STRUCTURES

Structure allows one to store together multiple objects of different type and size. Because structures vary in size, they must be stored into tokens. The 'TOKEN' command defines the tokens, 'EXP.MEM>' allocates them in expanded memory, and 'BECOMES>' and 'EQUIV>' assign an array to them.

All functions related to structures are found in 'STRUCTUR.ASY'. All recoverable errors are given an error number in the range 900-905 and a text describing the error is stored in '?ERROR"'. All errors are propagated.

The syntax and a short description of the main functions is shown below. Arguments may be: 'str' a structure (integer array), 'item#' the item in the structure to operate on, 'scalar', 'array' or 'string' the content of an item, or 't/f' a true/false flag.

{}CREATE [n — str]	To create a structure with 'n' items, initially empty.
{}DELETE [str item# — str]	To delete the content of an item.
{}SIZE [str item# — str size]	Return the size of item (# of elements in array or characters in string), or 0 if empty.

The following functions extract the content of an item as a scalar, an array or a string respectively. An error is returned if the request is inconsistent with the data stored.

{}READ [str item# — scalar]

{}READ[] [str item# — array]

{}READ" [str item# —] (— string)

but if the item is empty, a scalar '0' or a null string is returned. Additionally, the functions '{}READ?' and '{}READ[]?' are similar except that they also return a logical flag: *true* if there is data stored, and *false* if item empty, in which case, nothing is returned on stack.

The following functions store a scalar, an array or a string into a structure:

{}STORE [str scalar item# — str]

{}STORE[] [str array item# — str]

{}STORE" [str item# — str] (string — str)

4.3 WAVEFORMS

Several functions can be found in 'WFM.ASY' which manipulate waveforms. It is also possible to manipulate waveforms directly with the functions defined in the previous section and referring to Section 2.2.2 for the internal format of a waveform. In that case, direct manipulation should all be done or defined in few files, in case the definition of a waveform is altered later.

All recoverable errors are in the range 910-913 with a text stored in '?ERROR"' describing the error. Recoverable errors are not propagated; it is up to the caller to check if an error has occurred by testing '?ERROR#'.

The syntax of these functions is shown below, where 'wfm' represent a waveform:

`WFM.CREATE [ — wfm ]`	Create an empty waveform.
`WFM[WF] [ wfm — array ]`	Extract the raw data array from the waveform. It is only affected with the vertical gain and offset. Scalar '0' is returned if no data present.
`WFM[WFP] [ wfm — array ]`	Extract the processed data array, or scalar '0' if no data present.
`WFM[X.ARRAY] [ wfm y-arr — x-arr y-arr ]`	Extract the corresponding time array. The raw or processed data array must be obtained first.
`RAW [ — ]`	Set the raw data flag.
`PROCESSED [ — ]`	Set the processed data flag.
`WFM[WFM] [ wfm — array ]`	Extract the raw or processed data array, according to flag.

In addition, several utilities are defined in 'WFM.PGM' that perform operations directly on the raw or processed data arrays of waveforms as selected by 'RAW' and 'PROCESSED'. The three basic types of operations are: a monadic operation on a data array, a diadic operation between a data array and a scalar, and a diadic operation between two data arrays. In the later case, the time offset is used to align both data arrays and only the overlapping portions are kept, resulting in two arrays of same size. Therefore, it is very important to align both waveforms (see Section 4.5) before 'WFM.OP2' is called. In the descriptions below, 'fcn' refers to an ASYST word or a user-defined function. When operating on arrays, all arrays will have the same size.

`WFM.ENTER [ wfm — new-wfm ]`	Must be called to put the first waveform on stack and prepare the result.
`INSTALL fcn WFM.OP1 [ wfm — wfm ]`	Perform a monadic operation on the data array of a waveform. The syntax for 'fcn' is: `fcn [ array — newarray ]`
`INSTALL fcn WFM.OP1 [ wfm scalar — wfm ]`	Perform a diadic operation on the data array of a waveform and a scalar. The syntax for 'fcn' is: `fcn [ scalar array — newarray ]`
`INSTALL fcn WFM.OP2 [ wfm1 wfm2 — wfm ]`	Perform a diadic operation between the data arrays of two waveforms. The syntax for 'fcn' is: `fcn [ arr1 arr2 — new-arr ]`

WFM.CNT++	**[wfm -- wfm]**	Use the waveform scale factor field to implement a counter C (storing 1/C). This function increment C.
WFM.AVG.1st	**[wfm — avg-wfm]**	Start averaging.
WFM.AVG	**[avg-wfm wfm — avg-wfm]**	Add one more waveform to the average.

Here are few examples to illustrate how to properly used these functions. The first one shows how to negate a waveform:

```
wfm1  RAW WFM.ENTER   INSTALL NEG WFM.OP1
```

while this example shows how to add an offset to a waveform:

```
wfm1  RAW WFM.ENTER   1.2   INSTALL + WFM.OP1
```

and this last one shows how to subtract two waveforms (such as two samples obtained from differential channels or to compute an error term):

```
wfm1  RAW WFM.ENTER   wfm2  INSTALL -  WFM.OP2
```

4.4 FILES

The functions defined in file 'DAQ-FILE.ASY' allow one to store and recall waveforms as with the interactive function keys, ie. by specifying a file number. In addition, some more general functions are defined in file 'DAQ-VARR.ASY' to allow one to store or recall waveforms or any integer arrays in files specified by name. All these functions are not loaded permanently, but are precompiled into the 'DAQ-FILE.AOV' overlay. This overlay must be loaded as described at the beginning of this chapter before these functions can be used.

All recoverable errors are in the range 930-934, including recoverable file I/O errors (such as file not found, etc.). A short text is also stored in '?ERROR$'. Recoverable errors are not propagated; it is up to the caller to check if an error has occurred by testing '?ERROR#'.

The following two functions store or recall waveforms from files. 'item#' is a waveform index, 'file#' is as defined with the 'Ctl/F8' function key, and 'wfm' is a waveform:

```
DF.RECALL  [ item# file# — wfm ]

DF.STORE   [ wfm item# file# — ]
```

but if the item is empty, 'DF.RECALL' will return the scalar '0'.

The following functions allow one to open a file, read or write to it, and close it. Only single precision integer arrays (and hence waveforms) of any length can be stored. 'filnam' specifies the file. 'FILE.CLOSE' is used to close a file.

VA.CREATE	**[—] (filnam —)**	Create a new empty data file.

VA.OPEN [—] (filnam —)	Open a data file.
VA.STORE [array item# —]	Store the integer array into file.
VA.RECALL [item# — array]	Recall an array from the file. Nothing is returned if an error occurs; '?ERROR#' should be checked.

4.5 WAVEFORM DEJITTER

Three functions defined in file 'DAQ-DJIT.ASY' implement the waveform dejitter operation described in Section 3.2.8. An index offset is calculated using a least-square algorithm to obtain the best match between the data arrays of two waveforms. This index offset is converted to time offset and stored in the time offset field of the waveform. The 'RAW' and 'PROCESSED' functions are use to select the proper data for subsequent operations. Note that these functions release any transient overlay loaded.

DEJITTER.REF [wfm — wfm]	Store the reference array (raw or processed data).
REAL VECTOR[N , f_c] DEJITTER.FILTER :=	Store the pre-filter specification. Cleared by 'DEJITTER.CLEAR'.
DEJITTER.CLEAR [—]	Clear the reference array and release the memory.
DEJITTER.WFM [wfm — wfm]	Align the raw or processed data of the waveform against the reference.

The example below illustrates the use of the dejitter function for the averaging of two waveforms (note that 'WFM.AVG.1st' and 'WFM.AVG' could have been used instead):

```
wfm1  RAW DEJITTER.REF   WFM.ENTER
wfm2      DEJITTER.WFM   INSTALL + WFM.OP2   0.5 INSTALL * WFM.OP1
```

4.6 DIGITAL FILTERS

As shown before, waveform may include a data processing list, which may include digital filters. The two types of filters considered here are the '*finite impulse response*' (FIR) and '*infinite impulse response*' (IIR) filters. They are derived from the z-transform theory and discrete (or sampled) systems theory ([1], [2], [3]).

The FIR filter involves the multiplication of the previous values of the input sequence '{x}' with some coefficient to obtain the resulting sequence '{y}'. The resulting algorithm is non-recursive. The z-transform and the corresponding non-recursive equation is shown here:

$$H(z) = b_0 + b_1 \cdot z^{-1} + b_2 \cdot z^{-2} + \ldots + b_m \cdot z^{-m}$$

and

$$y_n = b_0 \cdot x_n + b_1 \cdot x_{n-1} + b_2 \cdot x_{n-2} + \ldots + b_m \cdot x_{n-m}$$

The IIR involves the multiplication of both the input and output sequences with some coefficients to obtain the result. The resulting algorithm is recursive: previous values of the output sequence are also used. The z-transform and the corresponding recursive equation is shown here:

$$H(z) = \frac{b_0 + b_1 \cdot z^{-1} + b_2 \cdot z^{-2} + \ldots + b_m \cdot z^{-m}}{a_0 + a_1 \cdot z^{-1} + a_2 \cdot z^{-2} + \ldots + a_k \cdot z^{-k}}$$

and

$$y_n = (b_0 \cdot x_n + b_1 \cdot x_{n-1} + b_2 \cdot x_{n-2} + \ldots + b_m \cdot x_{n-m} \ldots$$

$$\ldots - a_1 \cdot y_{n-1} - a_2 \cdot y_{n-2} - \ldots - a_k \cdot y_{n-k} \quad) / a_0$$

The realization of an IIR filter from the equations above often results in significant errors due to numerical round-off for higher order filters. These errors can be avoided by realizing the filter in a cascade form, ie. H(z) is defined as the product of small order filters:

$$H(z) = \frac{b_{00} + b_{01}z^{-1} + b_{02}z^{-2}}{a_{00} + a_{01}z^{-1} + a_{02}z^{-2}} \cdot \frac{b_{10} + b_{11}z^{-1} + b_{12}z^{-2}}{a_{10} + a_{11}z^{-1} + a_{12}z^{-2}} \cdot \ldots$$

In general, the filter coefficients '{a}' and '{b}' are computed for a given Δt. Computing these coefficients is not easy and often time consuming. Many techniques are available for computing the proper coefficients, such as the impulse invariance, the bilinear transformation, the impulse response windowing, and the frequency-sampling. It is possible to store multiple sets of pre-computed coefficients in a file and specify only the file name for further access (see Section 3.2.6). Each set correspond to a given Δt; thus the file can be seen as a lookup table of coefficients.

The following functions, found in file 'PROCESS.ASY', are used to build files containing FIR or IIR filter coefficients. The programs attempts to recover some errors, but all errors are propagated.

These two functions create a new empty file for FIR or IIR filters respectively:

CREATE.FIR.FILE [—] (filename —)

CREATE.IIR.FILE [—] (filename —)

while the next three functions are used to store the pre-computed coefficients ('{a}', '{b}' and '{a&b}') of FIR, IIR and cascaded IIR filters respectively into these files:

```
FIR>FILE    [ {b} ΔT — ]  ( filename — )

IIR>FILE    [ {a} {b} ΔT — ]  ( filename — )

IIR>FILE    [ {a&b} ΔT — ]  ( filename — )
```

where '{a}' and '{b}' are real vectors where the a_0 and b_0 are stored last; and '{a&b}' is a real matrix of dimension Nx6, where each line stores the coefficients of one cascade stage : [b_2 b_1 b_0 a_2 a_1 a_0].

4.7 INSTRUMENTATION CONTROL

Several functions are defined in 'SCOPE.ASY' which provide a standard procedure to interface with the various digitizers. They act as the main dispatcher to device specific routines. These functions are not permanently loaded, but are precompiled into the 'SCOPE.AOV' overlay, which must be loaded first.

All recoverable errors are in the range 920-929 or 496-543 (GPIB errors). Recoverable errors are not propagated; it is up to the caller to check if an error has occurred.

All these functions operation on a single instrument. This function selects a particular digitizer (specifying its channel #), and must be called first:

```
SCOPE.SELECT  [ ch# — ]
```

These three functions implement those accessed through the function keys 'Shf/F6', 'F6' and 'F7' respectively:

```
SCOPE.RESET  [ — ]

SCOPE.ARM  [ — ]

SCOPE.READ.WFM  [ df.wfm — wfm ]
```

where 'df.wfm' is the default waveform, or an empty waveform (create with 'WFM.CREATE'). To retrieve a default waveform entered with the 'Shf/F5' key, use the following lines:

```
CH.DEF CH# {}READ[]?
NOT IF  WFM.CREATE  THEN
```

4.8 PLOTTING GRAPHICS

Several functions are defined in the files 'PLOT.LIB', 'PLOTTER.UTL' and 'COLORS.UTL' to extend the plotting capabilities and provide hardcopy support to

various destinations. All these functions are also available by loading the 'PLOT.AOV' overlay.

The functions 'X.PLOT', 'XY.PLOT', 'X.DPLOT', 'XY.DPLOT' are defined to be used in place of 'X.AUTO.PLOT', 'XY.AUTO.PLOT', 'X.DATA.PLOT', 'XY.DATA.PLOT' respectively. They generate nicer plots and can be used to draw on screen or on hardcopy.

The functions defined in 'COLORS.UTL' are used to maintain two palettes of colors for the screen and the hardcopy (plotter), selected with 'USE.SCREEN.COLORS' and 'USE.PLOTTER.COLORS' respectively. The variable '#PENS' is set with the number of pens (ie. colors) to use, and the functions '1ST.COLOR', 'NEXT.COLOR' and 'SET.COLOR [n —]' are used in place of 'COLOR' to select a color for a curve. Previous colors will be reuse if needed.

The functions defined in 'PLOTTER.UTL' are used to redirect the plots to a hardcopy device or to a file (plots will therefore be suppressed on screen).

"PLOTTER.PORT [—] (dest —)	Specify the destination of hardcopy plots. Valid destinations are: '"NULL' to disable plotter, 'COM1:' or 'LPT1:' for serial or parallel port, 'PRN:' for the DOS spooler ('PRINT' command), 'filename' to send to files *.001, *.002, etc., or 'filename.ext' to accumulate all plots in a single file.
PLOT>PLOTTER	Redirect all future plots to the hardcopy device. Usually used in conjunction with 'USE.PLOTTER.COLORS'.
PLOT>PLOTTER.STOP	Resume plots to the screen. Usually used in conjunction with 'USE.SCREEN.COLORS'.
PLOTTER.PAGE	Force a new page or a new file.
CHAR.%SIZE [%h %v]	Specify character size, both horizontally and vertically. Default=1.

REFERENCES

[1] M. Dion, "*Design of Digital Signal Processing Algorithms for Enhancing the Measurements of Ultra-Fast Electromagnetic Transients*", DREO Report 1095, 1991

[2] A.V. Oppenheim and R.W. Schafer, "*Digital Signal Processing*", Prentice-Hall, 1975

[3] L.R. Rabiner and B. Gold, "*Theory and Application of Digital Signal Processing*", Prentice-Hall, 1975

[4] "*ASYST Version 3.10 Documentation*", Asyst Software Technologies, 1989

[5] "*7912HB Programmable Digitizer Manuals*", Tektronix, 1987

[6] "*SCD1000/SCD5000 Transient Digitizer Manuals*", Tektronix, 1990

[7] "*DSA 601 & DSA 602 Digitizing Signal Analysers Manuals*", Tektronix, 1989

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1. **ORIGINATOR** (the name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Establishment sponsoring a contractor's report, or tasking agency, are entered in section 8.)

Defence Research Establishment Ottawa
Ottawa, Ontario
K1A 0Z4

2. **SECURITY CLASSIFICATION** (overall security classification of the document including special warning terms if applicable)

UNCLASSIFIED

3. **TITLE** (the complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S,C or U) in parentheses after the title.)

Description of a Data Acquisition System for the Measurement of EMP Transient Fields (U)

4. **AUTHORS** (Last name, first name, middle initial)

Dion, M.

5. **DATE OF PUBLICATION** (month and year of publication of document)

March 1992

6a. **NO. OF PAGES** (total containing information. Include Annexes, Appendices, etc.)

40

6b. **NO. OF REFS** (total cited in document)

7

7. **DESCRIPTIVE NOTES** (the category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.

DREO Technical Note

8. **SPONSORING ACTIVITY** (the name of the department project office or laboratory sponsoring the research and development. Include the address.

Nuclear Effects Section
Defence Research Establishment Ottawa
Ottawa, Ontario K1A 0K2

9a. **PROJECT OR GRANT NO.** (if appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant)

Project 041LT

9b. **CONTRACT NO.** (if appropriate, the applicable number under which the document was written)

10a. **ORIGINATOR'S DOCUMENT NUMBER** (the official document number by which the document is identified by the originating activity. This number must be unique to this document.)

DREO TECHNICAL NOTE 92-7

10b. **OTHER DOCUMENT NOS.** (Any other numbers which may be assigned this document either by the originator or by the sponsor)

11. **DOCUMENT AVAILABILITY** (any limitations on further dissemination of the document, other than those imposed by security classification)

(X) Unlimited distribution
( ) Distribution limited to defence departments and defence contractors; further distribution only as approved
( ) Distribution limited to defence departments and Canadian defence contractors; further distribution only as approved
( ) Distribution limited to government departments and agencies; further distribution only as approved
( ) Distribution limited to defence departments; further distribution only as approved
( ) Other (please specify):

12. **DOCUMENT ANNOUNCEMENT** (any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). however, where further distribution (beyond the audience specified in 11) is possible, a wider announcement audience may be selected.)

Unlimited Announcement

UNCLASSIFIED

SECURITY CLASSIFICATION OF FORM

13. **ABSTRACT** (a brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual).

(U) This technical note describes a data acquisition system developed at DREO for the measurements of very fast transient electromagnetic fields generated during EMP testing. The system consists of several high speed transient digitizers, all connected to a personal computer. It has the capability to simultaneously record on several channels the response of single-shot events. The overall system is controlled by a master program called DAQ. DAQ can control a number of digitizers simultaneously, allowing automation of part of the measurement process. The data transferred to the PC is automatically bound with related information such as: experiment description, setup information (sensor, cable, attenuator, and instrument used), digitizer parameters, and data processing. DAQ can archive and retrieve the results of a large number of experiments. DAQ can produce plots of the results on the screen or on hardcopy, and also has the capability to import the plots into popular word processors for the preparation of reports. DAQ incorporates several digital signal processing routines, which may be used to process the raw data. Various implementations of Finite Impulse Response (FIR) and Infinite Impulse Response (IIR) filters are supported, as well as some of the classical approaches such as Butterworth low-pass and hi-pass filters.

Cette note technique décrit un système d'acquisition de données développé au CRDO pour permettre les mesures de champ électromagnétique extrêmement rapide qui sont générés lors des test d'IEM. Le système comprends plusieurs numériseurs très rapides reliés à un ordinateur personnel, permettant de contrôler plusieurs canaux simultanément et de permettre une certaine automatisation des opérations. Toutes les données obtenues des instruments sont combinées avec d'autres informations, telles que: une description de l'expérience, une description de l'arrangement des capteurs, câbles, atténuateurs et instruments utilisés, les paramètres des instruments, ainsi qu'une description des traitements numériques à effectués. Le système permet de gérer sur disque une large collection d'expériences. Il peut aussi afficher les résultats sous forme graphique à l'écran ou sur papier, ou encore sous forme compatible avec plusieurs logiciels de traitement de texte. Le système permet de définir des filtres numériques FIR ou IIR, de même que les approches classiques des filtres passe-bas ou passe-haut, telles que les filtres Butterworth.

14. **KEYWORDS, DESCRIPTORS or IDENTIFIERS** (technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus. e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus-identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

ASYST Programming Language
Data Acquisition
Digital Signal Processing
EMP Measurements
IEEE-488 (GPIR) Instrumentation Bus
Textronik 7912 Digitizers
Textronik SCD1000 Digitizers
Textronik DSA602 Digitizers
Transient Fields Measurements